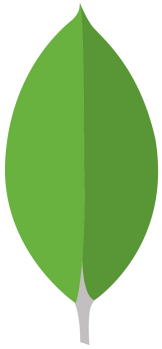
Sharding



mongoDB

Sharding

- DE las características más importantes e interesantes de Mongo.
- Puede traducirse como particionado, aunque Sharding es el concepto conocido.
- **MongoDB** utiliza esta técnica para gestionar la **carga de los servidores**. Distribuye los datos entre distintos ***shards*** (conjuntos de servidores que almacenan parte de los datos), para que la carga a la hora de realizar consultas e inserciones se reparta.

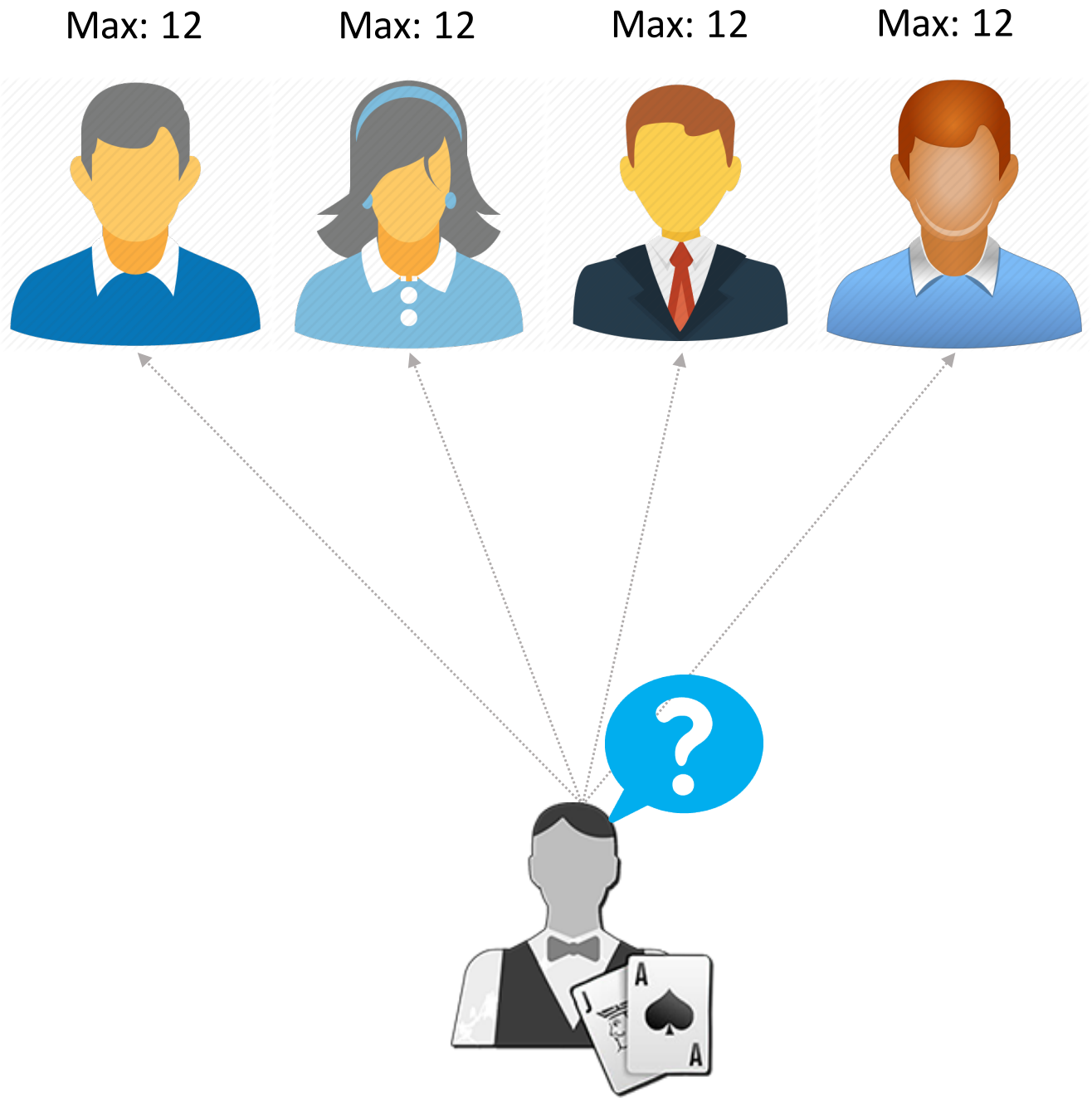
Concepto – Baraja de cartas

Kristina Chodorow

Vamos a jugar con una baraja de cartas española de 48 cartas. Para el que no la conozca, decir que esta baraja está dividida en cuatro palos: *oros, copas, espadas y bastos*. Cada palo tiene doce cartas numeradas del uno al doce.

Punto de partida

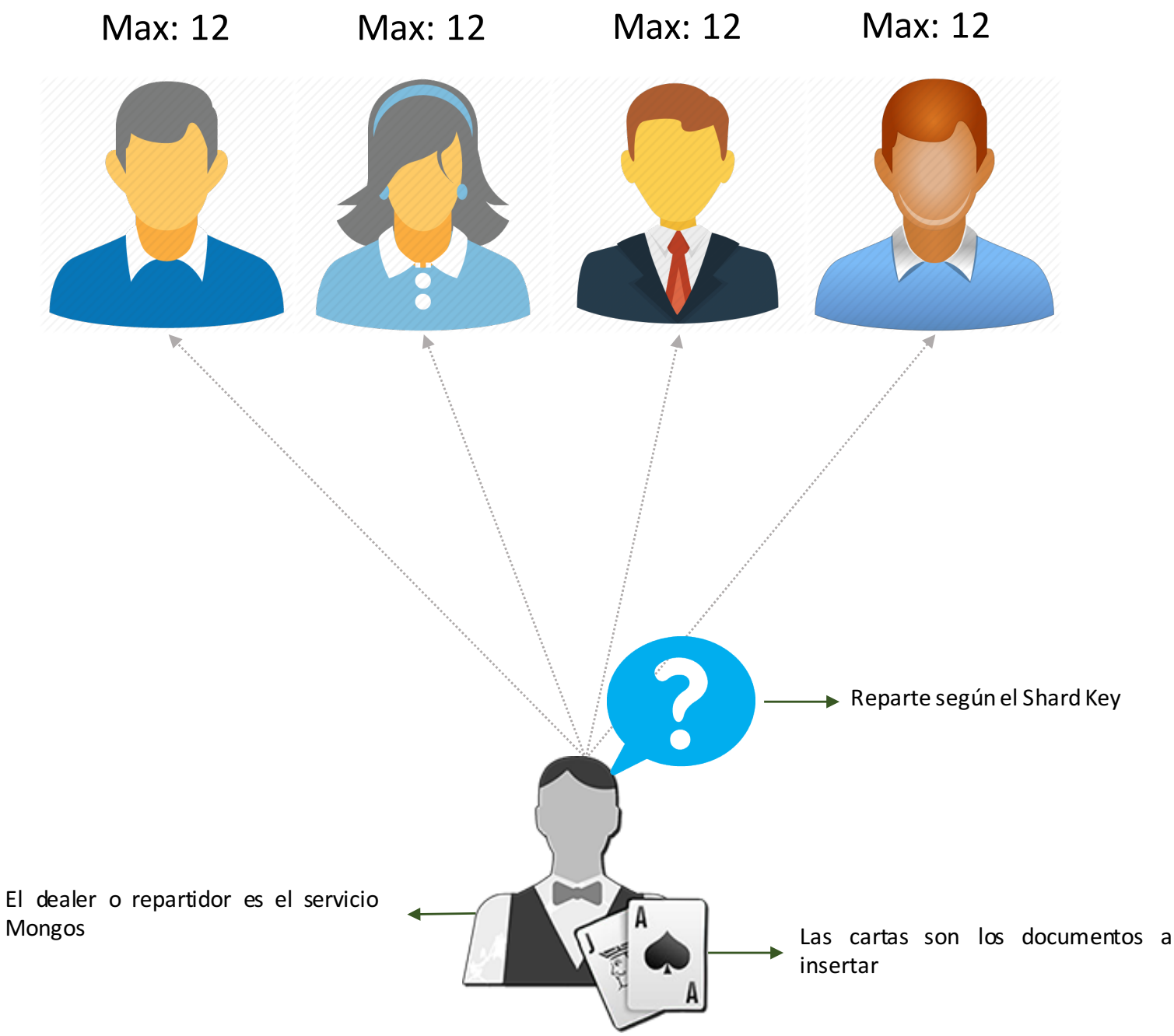
- En la partida jugarán cuatro jugadores, mientras que otra persona será la encargada de repartir las cartas. La persona que reparte las cartas no juega. Solo se dedica a repartir de una forma concreta: repartirá todas las cartas entre los cuatro jugadores, de manera que todos tengan doce cartas como máximo. Eso sí, antes de repartir y decidir a qué jugador le corresponderá la carta, mirará de qué carta se trata. Las cartas no son secretas ni deben esconderse.


Max: 12
Max: 12
Max: 12
Max: 12
?
A
A

¿En qué se parece este juego al *sharding* de MongoDB?

- Las cartas representarán los documentos a insertar, los jugadores representarán los *shards* y la persona que reparte se corresponde con el servicio *mongos*. Este servicio es el que conoce todos los *shards*, y es el encargado de repartir los documentos a partir de una **shard key** determinada.

Shards


Max: 12
Max: 12
Max: 12
Max: 12
Reparte según el Shard Key
El dealer o repartidor es el servicio Mongos
Las cartas son los documentos a insertar

Shard Key

- Elegir la clave por la cual repartir los documentos (**shard key**) es una de las decisiones más importantes que tendremos que tomar al usar *sharding* en **MongoDB**.

- Para el ejemplo vamos a suponer, que las cartas se repartirán según su valor. Es decir, nos fijaremos en el número que tenga la carta - del uno al doce - y repartiremos en consecuencia.

Empezando el juego

- La persona que reparte las cartas coge la primera del mazo y mira su valor. Es el tres de copas. El palo de la carta nos da igual, solo nos fijamos en el valor. Como ninguno de los jugadores tiene cartas, se la da al primer jugador.

- Desde el punto de vista de **MongoDB**, podríamos decir que se se intenta insertar un documento y el proceso *mongos* decide que como todos los *shards* están vacíos, debe darle el documento al primero.

...

- Se repite hasta repartir la carta #12 al primer jugador, que llega al máximo, entonces el repartidor se da cuenta que los otros jugadores no tienen cartas y divide las cartas repartidas.
- Supongamos que las cartas repartidas eran: *3,4,9,10,9,2,3,1,12,5,5,6*
- *Ordenandas:*

1,2,3,3,4,5,5,6,9,9,10,12

Tomando en cuenta la Shard Key

Max: 12

Jugador 1:
1,2,3,3,4,5,5

Max: 12

Jugador 2:
6,9,9,10,12

Max: 12

Jugador 3:

Max: 12

Jugador 4:

… continúa repartiendo

- Se reparte una a una las cartas, cuando un jugador (Shard) llega nuevamente a 12 (máximo) se debe hacer la partición.
- Supongamos que se repartieron estas cartas:

1,4,5,6,7,8,8,9,10,11,11,12.

Está en bloques de 12 para el ejemplo, pero se hace una a una hasta que un jugador llegue al máximo.

Tomando en cuenta la Shard Key

Max: 12

Jugador 1:
1,1,2,3,3,4,4

Max: 12

Jugador 2:
5,5,5,6,6,7,8,8

Max: 12

Jugador 3:
9,9,9,10,10,11,11,12,12

Max: 12

Jugador 4:

Penúltimo reparto de 12*

Vamos por el penúltimo reparto de cartas, que ya ordenado es: 3,4,5,6,6,7,7,8,9,10,11,12. Y repartido entre los jugadores sería algo así como:

- **Jugador 1**: 1,1,2,3,3,3
- **Jugador 2**: 4,4,4,5,5,5,5,6,6,6,6
- **Jugador 3**: 7,7,7,8,8,8,9,9,9,9
- **Jugador 4**: 10,10,10,11,11,11,12,12,12

Finalizando el juego

Tras el último reparto de doce cartas, al final las cartas de los jugadores quedarían así:

- **Jugador 1**: 1,1,1,1,2,2,2,2,3,3,3,3
- **Jugador 2**: 4,4,4,4,5,5,5,5,6,6,6,6
- **Jugador 3**: 7,7,7,7,8,8,8,8,9,9,9,9
- **Jugador 4**: 10,10,10,10,11,11,11,11,12,12,12,12

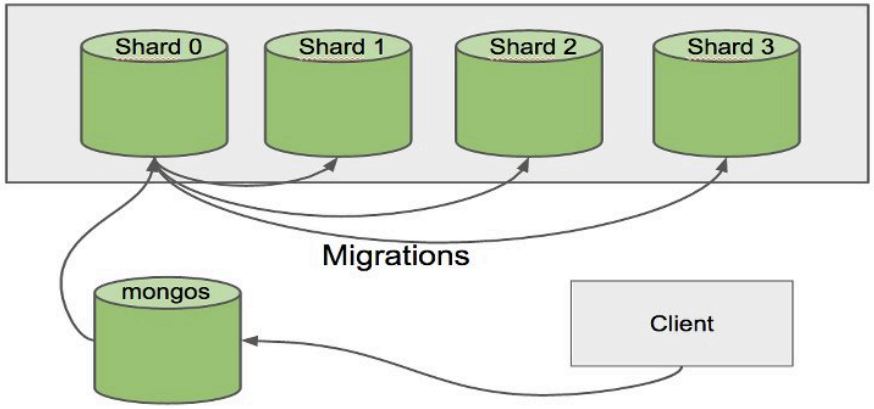
El tamaño de las particiones*

- En **MongoDB**, el tamaño de las particiones no se mide en número de documentos, si no en MB.
- Por defecto un *shard* tiene un tamaño máximo de 64 MB, aunque es algo que podemos configurar.
- Se cambiaría para lograr disminuir la cantidad de movimientos entre shards o los documentos se acumulan en pocos shards.

¿Qué es un chunk en MongoDB?

Ya sabemos que MongoDB es capaz por sí mismo de mantener nuestro cluster balanceado para una o varias colecciones.

Lo hace en base al número de chunks que guarda cada shard y no en base al número de documentos ni al tamaño ocupado por ellos.



¿Qué es un chunk?

- El shard key aceptará unos valores, desde un valor mínimo a un valor máximo. Ese rango lo dividiremos de tal forma que un shard guardará los datos asociados a una parte del rango, otro shard guardará otra parte, y así sucesivamente.
- El concepto de chunk es abstracto, no son datos en sí mismos, y está referido a cada uno de los rangos en que dividimos los datos.
- Esta información (metadata) se guarda en la configdb, que es la base de datos que gestionan los config servers.

Ejemplo

Chunk _id	Valor mínimo	Valor máximo	Shard
“_id” : “testdb.presplit-x_7000.0”	“min” : { “x” : 7000 }	“max” : { “x” : 8000 }	“shard” : “shard0001”
“_id” : “testdb.presplit-x_8000.0”	“min” : { “x” : 8000 }	“max” : { “x” : 9000 }	“shard” : “shard0001”
“_id” : “testdb.presplit-x_9000.0”	“min” : { “x” : 9000 }	“max” : { “x” : 10000 }	“shard” : “shard0002”
“_id” : “testdb.presplit-x_10000.0”	“min” : { “x” : 10000 }	“max” : { “x” : 11000 }	“shard” : “shard0002”

¿Está balanceado nuestro sistema?

Chunks	Diff
Fewer than 20	2
20 – 79	4
80 and greater	8

- Supongamos que, por ejemplo, tenemos 2 shards y nuestra colección está dividida en 31 chunks. Si el shard0000 tiene 17 chunks y el shard0001 tiene 14 chunks, MongoDB considera el sistema balanceado al ser la diferencia menor a 4 chunks.

Funcionamiento de MongoDB

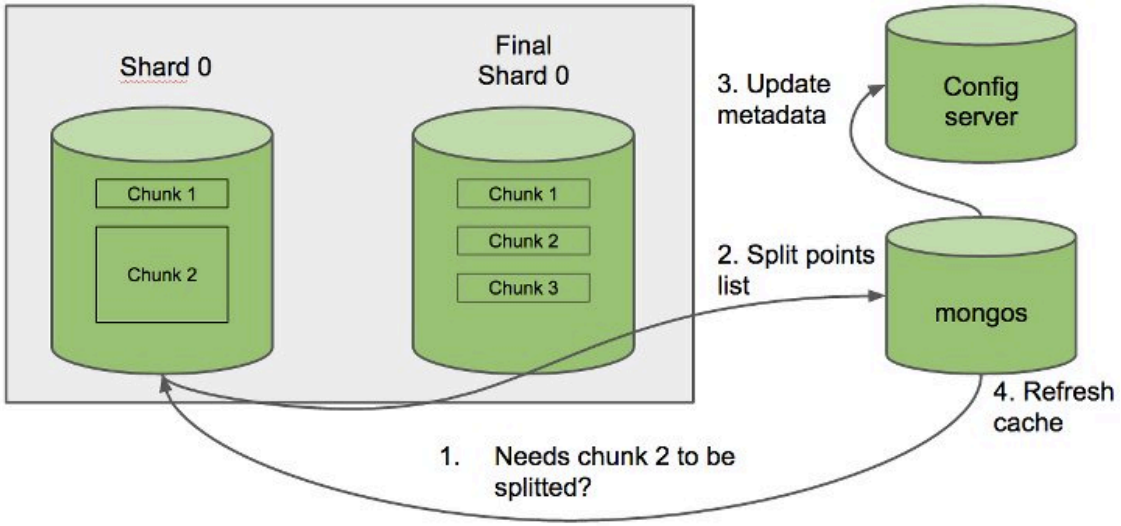
- Cuando creamos una colección por primera vez MongoDB la asigna al shard donde reside la base de datos a la que pertenece, y crea un chunk que recoge todo el rango de valores posibles para la shardkey escogida.

```
testdb.exampleCollection
shard key: { "x" : 1 }
chunks:
shard0000 1
{ "x" : { "$minKey" : 1 } } -->> { "x" : { "$maxKey" : 1 } } on : shard0000 Timestamp(1, 0)
```

Algoritmo

1. Según vamos insertando datos en la colección nuestro chunk va creciendo en tamaño (su tamaño máximo por defecto es de 64MB) hasta llegar al valor máximo establecido. En ese momento MongoDB realiza el split, es decir lo divide en dos.
2. Ahora MongoDB comprueba si el sistema está balanceado. Si no lo está lo que hace es mover un chunk del shard que más tenga al que menos (chunk migration).

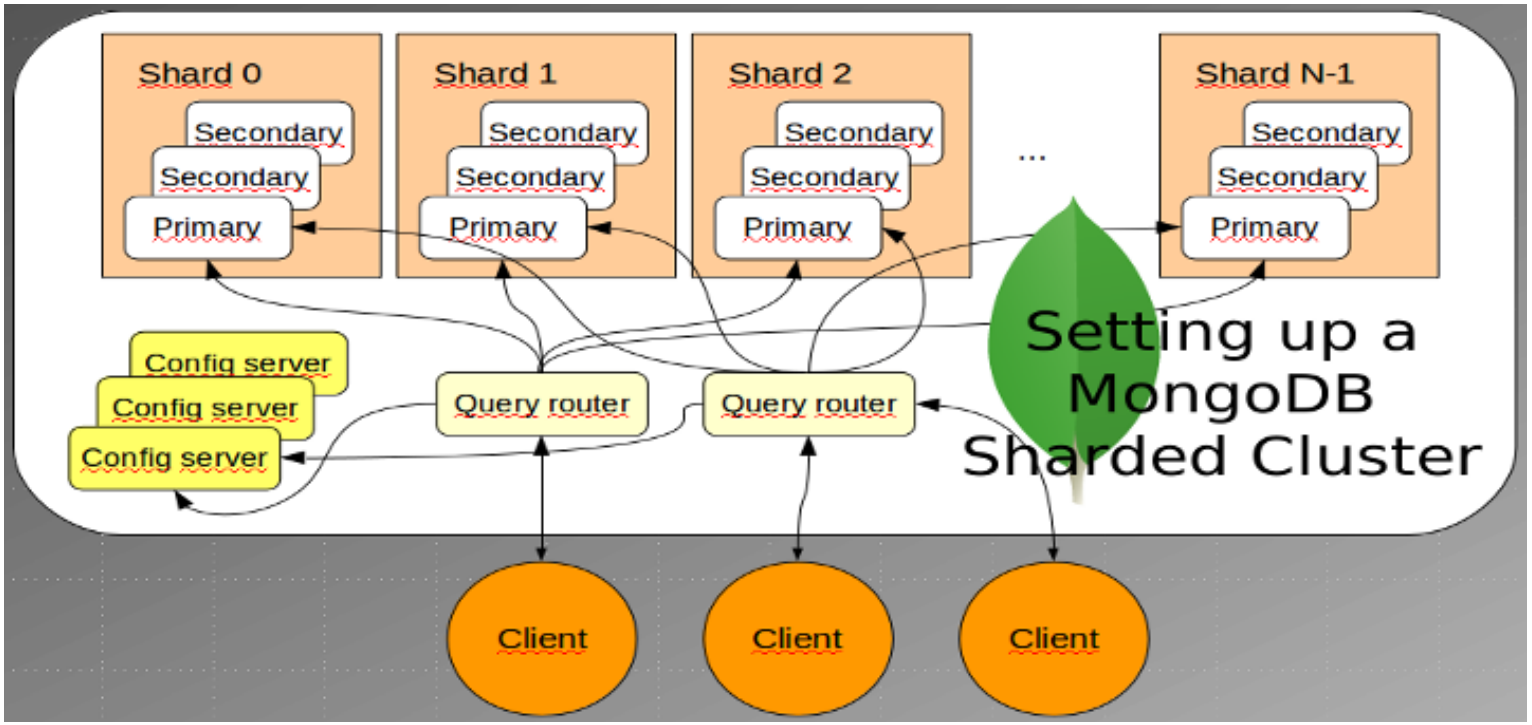
Algoritmo



1. Todos los mongos guardan información del tamaño de todos los chunks de cada uno de los shards. Cuando un mongos detecta que un chunk está recibiendo muchas escrituras pregunta al shard donde este reside si necesita ser dividido.
2. El shard le devuelve una lista con los puntos de división. Si está vacía quiere decir que el chunk no ha crecido aún lo suficiente y el mongos no hace nada. En caso contrario la lista contiene aquellos puntos por los que el chunk se podrá dividir.
3. El mongos escogerá uno de ellos y actualizará la información en la base de datos del config server.
4. Por último, el mongos actualizará su caché.

- La colección changelog, de la configdb, guarda registro de todos los split que se realizan. En el siguiente ejemplo se ve cómo el chunk que contiene los documentos cuyo rango de la shardkey comprende desde el valor 0 al máximo se divide en dos, desde el 0 al 49999 y desde el 50000 al valor máximo.

CASO: sharding cluster



http://www.mongodbspain.com/es/2015/01/26/how-to-set-up-a-mongodb-sharded-cluster/